NASA/TM—2007-214840

# Development and Application of a Portable Health Algorithms Test System

*Kevin J. Melcher*
*Glenn Research Center, Cleveland, Ohio*

*Christopher E. Fulton, William A. Maul, and T. Shane Sowers*
*Analex Corporation, Brook Park, Ohio*

July 2007

# NASA STI Program . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI Program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI program provides access to the NASA Aeronautics and Space Database and its public interface, the NASA Technical Reports Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or cosponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include creating custom thesauri, building customized databases, organizing and publishing research results.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at *http://www.sti.nasa.gov*

- E-mail your question via the Internet to *help@sti.nasa.gov*

- Fax your question to the NASA STI Help Desk at 301–621–0134

- Telephone the NASA STI Help Desk at 301–621–0390

- Write to:
  NASA Center for AeroSpace Information (CASI)
  7115 Standard Drive
  Hanover, MD 21076–1320

NASA/TM—2007-214840

# Development and Application of a Portable Health Algorithms Test System

*Kevin J. Melcher*
*Glenn Research Center, Cleveland, Ohio*

*Christopher E. Fulton, William A. Maul, and T. Shane Sowers*
*Analex Corporation, Brook Park, Ohio*

Prepared for the
54th Joint Army-Navy-NASA-Air Force (JANNAF) Propulsion Meeting, 5th Modeling and Simulation Subcommittee, 3rd Liquid Propulsion Subcommittee, 2nd Spacecraft Propulsion Subcommittee Joint Meeting
sponsored by the JANNAF Interagency Propulsion Committee
Denver, Colorado, May 14–17, 2007

National Aeronautics and
Space Administration

Glenn Research Center
Cleveland, Ohio 44135

July 2007

## Acknowledgments

The authors gratefully acknowledge Robert Button and Marcello Gonzales from the Glenn Research Center (GRC) Advanced Electrical Systems Branch for providing access to the Power Distribution Unit (PDU) test-bed, and for providing technical expertise during integration of the Portable Health Algorithms Test (PHALT) system with the PDU test-bed. The authors also express thier appreciation to the NASA Glenn Reseach Center Launch Systems Project Office for supporting the work presented in this paper.



*Level of Review*: This material has been technically reviewed by technical management.

Available from

NASA Center for Aerospace Information
7115 Standard Drive
Hanover, MD 21076–1320

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161

Available electronically at http://gltrs.grc.nasa.gov

# Development and Application of a Portable Health Algorithms Test System

Kevin J. Melcher
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

Christopher E. Fulton, William A. Maul, and T. Shane Sowers
Analex Corporation
Brook Park, Ohio 44142

## Abstract

This paper describes the development and initial demonstration of a Portable Health Algorithms Test (PHALT) System that is being developed by researchers at the NASA Glenn Research Center (GRC). The PHALT System was conceived as a means of evolving the maturity and credibility of algorithms developed to assess the health of aerospace systems. Comprising an integrated hardware-software environment, the PHALT System allows systems health management algorithms to be developed in a graphical programming environment; to be tested and refined using system simulation or test data playback; and finally, to be evaluated in a real-time hardware-in-the-loop mode with a live test article. In this paper, PHALT System development is described through the presentation of a functional architecture, followed by the selection and integration of hardware and software. Also described is an initial real-time hardware-in-the-loop demonstration that used sensor data qualification algorithms to diagnose and isolate simulated sensor failures in a prototype Power Distribution Unit test-bed. Success of the initial demonstration is highlighted by the correct detection of all sensor failures and the absence of any real-time constraint violations.

## Introduction

The development of on-board, real-time diagnostic systems and algorithms is challenging. The diagnostic software must satisfy real-time performance metrics and show relevance to the targeted platform. Testing and evaluation are required throughout the development process up to final implementation. As the diagnostic system develops, the evaluation process must also advance from software simulation to hardware implementation and can range from initial feasibility studies to final verification and validation testing. While much effort has addressed the development and testing of these algorithms within a software environment, there is a gap in transitioning these technologies to relevant hardware platforms where they can begin to establish credibility. The PHALT System was envisaged as a means of filling this gap.

Initially, development of the PHALT System was motivated by the need to conduct real-time hardware-in-the-loop feasibility studies of sensor data qualification algorithms (ref. 1) for NASA's new crew launch vehicle, Ares I. However, researchers at the NASA Glenn Research Center working to meet this need soon realized that a broader capability would prove useful in addressing issues faced by the system health management community. System hardware designs are often nearing their final stage by the time health management needs are considered. Adding additional instrumentation or changing designs late in the design process in order to meet diagnostic and prognostic goals can result in significant increases in time and cost. These impacts might be mitigated by a PHALT System that allows the implementation of low-level health management solutions at the component or subsystem level during trade studies when designs can be changed without high cost. Another issue with system health

management is a perceived lack of credibility on the part of some system designers. This “trust gap” can be overcome by providing a platform that allows health management solutions to seamlessly evolve with the system design—moving from component to subsystem application, and from a simulation-based context to real-time hardware-in-the-loop operation with a relevant test-bed. Additionally, there is a desire to develop a system that can be replicated inexpensively, providing a capability, not just for Glenn Research Center, but for NASA as a whole. With these issues in mind, GRC researchers began to develop a system that would meet the short-term need while pursuing the goal of a broader capability.

The objective of this paper is to describe the function and purpose of the PHALT System and to demonstrate its potential. The paper first describes a functional hardware and software architecture for the PHALT System. This architecture is followed by a discussion of the preliminary implementation and demonstration. For the demonstration, a sensor data qualification method (i.e., diagnostic algorithm) is applied to an electrical power distribution unit test-bed. The preliminary implementation was evolved starting with test data playback in a non-real-time software environment and moving to real-time hardware-in-the-loop testing. Results of this demonstration are also discussed. The paper concludes with a summary of the work.

## PHALT System Functional Architecture

As a first step in developing the PHALT System, a functional architecture was conceived to guide the selection of hardware and software (see fig. 1). This architecture has four primary functions: obtain data from the test article, analyze the data, act on the data, and manage system operation.

Data can be obtained from one of three sources depending on the stage of test article development: mathematical model, test data playback, or real-time (R-T) data acquisition. A mathematical model is often available early in the design process and is useful for initial algorithm development. Test data can be played back in a time synchronized fashion and is useful for refining diagnostic system implementation prior to hardware-in-the-loop testing. Real-time data acquisition is required for hardware-in-the-loop testing and may include a variety of analog and digital sources. Note that the Math Model and Data Acquisition modules can be used for open- and closed-loop applications, while the Test Data module is useful only for open-loop.

Once data is acquired by the PHALT System, it can be processed through a series of health assessment functions that the developer is interested in evaluating. The Data Conditioning, Failure Simulation, and Diagnostic/Prognostic modules shown in figure 1 categorize these functions. In the Data Conditioning module, data is modified (e.g., converted from raw data to engineering units, re-sampled,

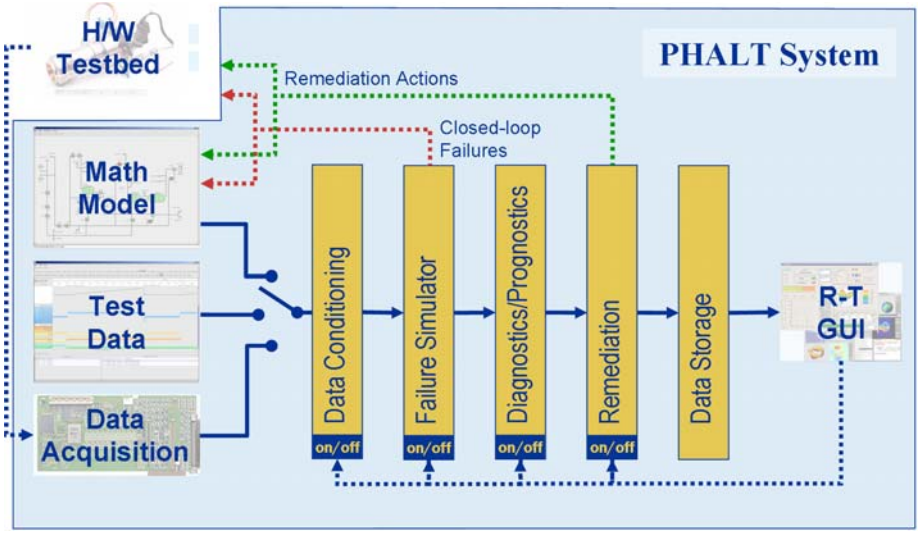


Figure 1.—PHALT System functional architecture.

or filtered) to ensure that a consistent data set is available to the other analysis modules. The Failure Simulation module can superimpose failures on nominal system data when the failing of actual test hardware is not convenient, desirable, or possible. The Diagnostic/Prognostic module is the heart of the PHALT System, providing a mechanism for defining and testing any number of health management algorithms that may be available to the user. In this architecture, health management algorithms can be evaluated individually or as an integrated package.

After analyzing the data, the system is able to act on the outcome. Any simulated faults that are part of a closed-loop system, must necessarily be fed back to the test article to insure a representative system response to the failure. Also, given a failure diagnosis, it may be desirable to act in a way to remediate or mitigate the failure. The Remediation module provides a mechanism for selecting appropriate remediation strategies and communicating them to the test article (i.e., test-bed or math model).

System operation is maintained through the use of a real-time graphical user interface (GUI). This GUI is used to load pre-defined test scenarios; to initialize the system for a given test; to run test scenarios; and to upload data from the Data Storage module for further analysis. The Data Storage module provides a consistent and organized means for storing data generated by the PHALT System so that it can be accessed for future analysis, verification, and reporting.

The envisaged architecture was implemented in a modular fashion via an extensible, rapid prototyping, graphical programming environment to provide a “plug and play” like interface for implementing the fault detection, isolation, diagnostic/prognostic, and remediation algorithms that are part of health assessment technologies. In addition to graphically programmed algorithms, the architecture supports the use of legacy code to the greatest extent possible. It also provides an integrated, relatively inexpensive, hardware/software solution with a seamless path from simulation-based development to real-time, hardware-in-the-loop demonstration.

## PHALT System Hardware and Software

The next step in developing the PHALT System was the definition of lower level requirements to guide the selection of the hardware. A variety of issues were considered in developing these requirements. First, the development function had to be separated from the real-time hardware-in-the-loop function, so that only the real-time system would be integrated with the test-beds. Since it was likely that early testing would reveal weaknesses in some of the algorithms, a process was required whereby the algorithms could be safely revised and tested off-line, thus avoiding potential damage to any test article. To accomplish that objective, data acquisition would only be made available to the real-time system. Second, a portable system was desired as potential test facilities would be spread not only throughout various buildings across GRC, but throughout various NASA Centers as well. Third, the PHALT System required a significant storage capacity to accommodate data sets captured during long (i.e., hours) tests with sampling rates that are nominally in the range of 25 to 50 Hz, but may be higher. Fourth, during testing, the real-time system had to be able to operate unattended in a facility environment for long periods of time.

### Hardware/Software Selection

Several different hardware/software solutions were considered for implementing the PHALT System architecture. A personal computer-based approach (see fig. 2) using Simulink, Real-time Workshop, and xPC Target software tools from the Mathworks Inc. was selected based on a balance of cost, capability, and implementation schedule. Simulink software provides a graphical programming environment for time-based simulation that is extensible and can be implemented in a modular fashion. Legacy FORTRAN or C code can be incorporated, as well as, third party application libraries supporting data acquisition devices, shared memory applications, etc. Adding Real-time Workshop and xPC Target software to the architecture provided the seamless path from non-real-time development to real-time execution. And, the bootable, real-time executable it generates is targeted for PC hardware, so

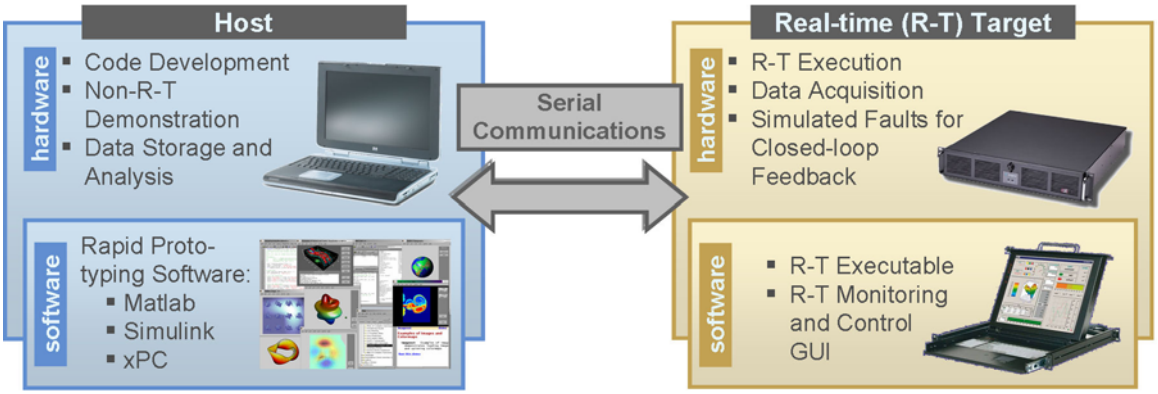


Figure 2.—PHALT System hardware and software.

hardware costs are minimal. Hardware costs are also kept low by the fact that a single Matlab software license and development workstation can be used to support multiple real-time targets.

A high-end laptop was purchased for the development workstation (host system). The laptop uses the Windows XP (Microsoft Corporation) operating system, as xPC software is not supported on UNIX/LINUX type machines. To support programming in a graphical programming environment, the laptop has two (one built-in and one external) 17-in. diagonal displays. A large hard drive supports the storage and analysis of potentially large amounts of test data. Also, a variety of communication options are available: a serial port, several USB ports, and built-in wired and wireless Ethernet.

An industrial PC (real-time target) was purchased for real-time hardware-in-the-loop testing. The real-time target has a 3.0 GHz processor, 1 GB of memory, and a 250 GB hard disk. A rack-mountable keyboard/monitor provides access to the real-time target for system administration and file transfer. This system also has a variety of communication options: a serial port, a parallel port, and several USB ports.

The primary communications between the host and real-time target is currently via serial cable. The target computer is the embedded operational system with a real-time kernel. The fundamental process is to develop a model in Matlab/Simulink on the host computer. When ready, the model is compiled using automatically generated code and a C compiler. The target computer is booted up with the real-time kernel; and the compiled model is downloaded from the host to the target. During execution, a communication link can exist between the host computer and target computer. Host commands and changes to parameter values can be passed to software running on the real-time target. Target data can also be passed back to the host in real-time for monitoring purposes.

Data acquisition functions reside with the real-time target and need to be flexible since both the number and types of sensors required for testing algorithms can vary. The xPC software provides drivers for several readily available analog to digital conversion (ADC) boards. For general application, a board was selected that samples up to 64 single-ended data channels (32 channels in differential mode) with a resolution of 16 bits over 10 software selectable input ranges. The board also includes eight digital channel I/O ports and two analog output ports. In its current instantiation, CAN bus and Ethernet communications are also available as data acquisition options.

## Hardware/Software Integration

After the PHALT System hardware and software were purchased, GRC researchers began the task of implementing the elements of the architecture shown in figure 1. Initial efforts focused on determining the actual functionality of the integrated hardware and software. Then a Simulink block diagram was created for a single data channel. Aspects of that effort are reported here for the benefit of other researchers.

The xPC software allows the user to boot either with a bootable floppy disk (or equivalent device), or to boot the application directly from the hard drive. The first method allows communication with the host system and was the method used throughout this initial development. The second method was tested in

house to verify that it was an available option for future use when the target system may need to run unattended.

Rarely will the initial integration of software and hardware proceed without some complications and this task was no exception. During the initial tests, it was discovered Ethernet was not a workable solution for transferring information between the xPC host and real-time target computers. An incompatibility between the mother board and Ethernet card in the real-time target prevented the real-time kernel from booting. In fact, an Ethernet card that was compatible with the xPC software and with the mother board in the real-time target was not available. Therefore serial communications, an alternate, somewhat slower approach, was used to transfer information between the host and target machines.

Another problem that required trouble shooting involved the hard drive. The target system's hard drive was not recognized by the real-time kernel and therefore, would hinder real-time archiving of test-bed data. This was despite the fact the hard drive partitions were formatted as FAT16, per the xPC user's manual. A modification of the BIOS setup, changing the PATA/IDE configuration from "enhanced" (i.e., SATA) to "legacy," was required to resolve the problem.

A third problem was an inability to read a signal fed into the data acquisition card with the xPC software. The problem was solved with the help of Mathworks technical support, which sent a different version of the driver for the card.

Having resolved the hardware/software integration issues, a preliminary single-channel open-loop implementation of the PHALT System architecture was created using Matlab/Simulink/xPC software. The block diagram for this implementation is shown in figure 3. The "Data Source Indicator" is a Simulink constant block that is used to specify one of the five user selectable data sources: 1-data acquisition card, 2-user provided system model, 3-test data played back in real-time, 4-CAN bus card, or 5-Ethernet card. Raw data is obtained by the "Data Source Selection" block using the specified data source. This raw data is sent to two blocks, "Data Conditioning" and "Data Out." All archival data is stored by the "Data Out" block which internally uses xPC "File Scope" blocks. Both raw data and processed data with failures inserted are stored by the "Data Out" block. Either of these data sets can subsequently be used during playback mode. Any special processing of the raw data, such as conversion to engineering units, occurs in the "Data Conditioning" block. The processed data stream is then fed to the "Failure Simulation" block where one or more of several failure types may be superimposed on it. Failure models currently include: step to open, step to zero, drifts, intermittent, and noise. The failures can be inserted either concurrently or sequentially on different channels or on the same channel. The resulting output of the "Failure Simulation" block is then sent to the "Diagnostics" block where user specified diagnostics algorithms can be applied to the data stream. For this initial implementation, sensor data qualification algorithms were implemented. The "Diagnostics" block output data stream is also stored by the "Data Out" block for analysis by the user. Finally, any subset of the data stream can then be selected and presented to the user via the "R-T GUI" block which provides a real-time graphical user interface (GUI). The real-time GUI may reside either on the target machine's display or, if connected to the host, on the host's display. For this initial implementation, all functionality contained in the "Data Conditioning," "Failure Insertion," and "Diagnostics" blocks was implemented as Matlab S-functions

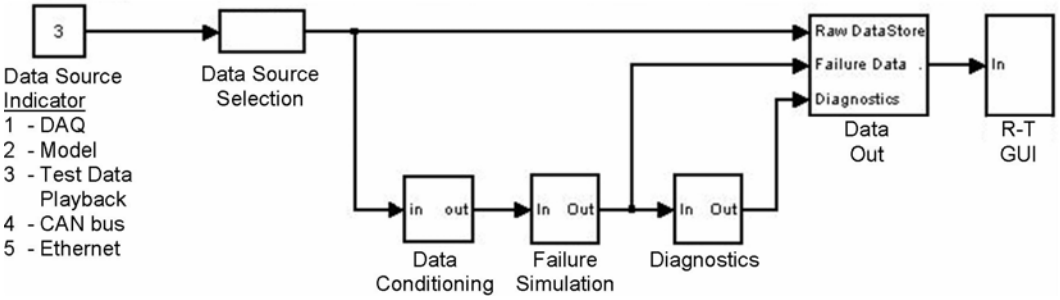


Figure 3.—Block diagram for preliminary PHALT System implementation in Simulink.

with custom C-code. An S-function is a segment of code that is structured and compiled into a Simulink library following a well-defined protocol. This initial implementation approach was required due to the use of legacy code, schedule constraints, and some lack of required functionality within Simulink. A more general and reconfigurable implementation approach is planned for future generations of the code.

Run-time parameters for the S-functions were defined with the use of Matlab data structures. Using these structures, a single variable—used to pass multiple parameter values to the S-functions. These parameters become part of the source code when xPC compiles the code into a downloadable real-time model. Furthermore, these parameters can be modified from the host machine prior to running the model, providing the user an ability to redefine test scenarios without recompiling the software.

One previously unforeseen benefit of the choice of hardware and software was that a software script could be used to automatically run a series of tests. Although this capability in and of itself may seem insignificant, the fact that this script was able to change failure modes and insertion times shows the utility of the PHALT System to develop, modify, and test various algorithms against a variety of failure scenarios in a reasonably short time frame. Another benefit is that the PHALT System can be used to conduct studies aimed at characterizing real-time performance of the code.

## PHALT System Demonstration

NASA's new Ares I crew launch vehicle provided an immediate application for the newly implemented PHALT System. Under the Ares I Program, GRC researchers are conducting concept feasibility studies to demonstrate the benefit of advanced sensor data qualification (SDQ) algorithms for Ares I Upper Stage systems. Advanced SDQ is a sensor diagnostic approach that utilizes available analytical redundancy and a variety of statistical methods to assess the health of sensors in a given system. As a means of demonstrating the advanced SQD algorithms, the PHALT System was used to monitor a prototype Power Distribution Unit (PDU) test-bed that is representative of a launch vehicle PDU. This demonstration was conducted in three phases: non-real-time evaluation using test data playback, real-time evaluation using test data playback, and real-time hardware-in-the-loop evaluation. Results from phase 1 are reported in reference 1. Results from phases 2 and 3 are described in this section. Here, implementation specific details of the demonstration including the diagnostic algorithm, the PDU test-bed, test-bed/PHALT System integration, and evaluation of the diagnostic results are discussed.

### Diagnostic Algorithms

Diagnostic algorithms embodied in the SureSense Data Quality Validation Studio (DQVS) were used by GRC researchers to demonstrate sensor data qualification (SDQ) for Ares I. DQVS is a commercial software product developed by Expert Microsystems in conjunction with GRC. SureSense automates the production of online data validation modules that detect sensor failures and other data anomalies. This technology has been applied in the aerospace, medical, chemical and nuclear industries (refs. 2 to 6).

SureSense's model-based validation algorithm combines limit filters and analytical redundancy with Bayesian decision logic. Limit filters detect gross sensor faults (e.g., open circuits). Analytical redundancy is a technique that predicts the value of a signal using values of other signals and known or empirically-derived mathematical relations. System correlations and signal redundancies provide many of these relations.

For real-time applications, the SureSense development environment also provides an automatic code generator, which converts the sensor data validation modules into run-time modules that can be integrated into control and monitoring systems. Run-time modules consist of the real-time kernel and the process model. The real-time kernel is the general-purpose signal validation engine while the process model defines the application specific configurations for the real-time kernel.

The SureSense real-time kernel was converted into a Simulink S-function for incorporation into the xPC Target environment. After compilation, the S-function version of the code was accessible in Simulink. This functionality allows heritage software applications to be incorporated into Simulink and

improves execution speed since the code is compiled rather than interpreted. The SureSense real-time kernel was translated into the S-function format with the sensor data validation modules created by the SureSense automatic code generator. These two modules were made into an S-function for the xPC Target system.

## Power Distrubution Unit Test-Bed

To demonstrate the feasibility of SDQ algorithms for Ares I Upper Stage systems, the PHALT System with embedded SDQ algorithms was applied to a prototype PDU test-bed. Constructed by the GRC Advanced Electrical Systems Branch, the test-bed was first used to generate data for training and validating the SDQ algorithms. A schematic of the PDU test-bed is shown in figure 4.

The PDU test-bed contained 6 relays—3 input relays and 3 output relays. The input and output relays were connected to a common bus. Each input relay could be connected to a separate power supply and each output relay could be connected to a separate, variable load. For the PDU architecture shown, only one power supply was available at any time. So, for this demonstration power supply “A” was used to represent the active power supply. Also, only two of the three loads shown were used. This resulted from a reconfiguration of the test-bed between Phases 2 and 3. The supply source provided DC voltage from 28.3 to 28.7 VDC continuously. Load “B” was cycled between 15 and 20 A; and load “A” was held constant at 15 A. Operating modes were defined based on the state of the active loads.

Figure 5 shows the primary operating modes and the associated instrumentation. In that figure, RinA, VinA, IinA, and VbusinA, respectively represent the state of input relay “A,” the voltage across input relay “A,” the current flowing through input relay “A,” and the voltage across the point where relay “A” attaches to the power bus. Measurements for the output relays are tagged in a similar fashion. Color is used to identify modes during which a given sensor is active. Only modes 1 and 2 were used during phase 3 testing. Relay configurations for the two operating modes are specified in table 1. Load “B” was active during mode 1, while both loads “A” and “B” were active during mode 2.

Each relay has a dedicated input or output voltage sensor and through current sensor. The bus has additional voltage sensors at each of the six (6) relay connections. Additional sensor data values used but not qualified by the SDQS are the relay on/off discretes. These discrete variables were generated by the PDU test-bed software and reflected the state of the designated relay, “0” for open and “1” for closed. An additional discrete variable was added in post-processing and indicated the combined output current state for the PDU. Table 2 shows possible values for this variable. Each value represents a different sub-mode.

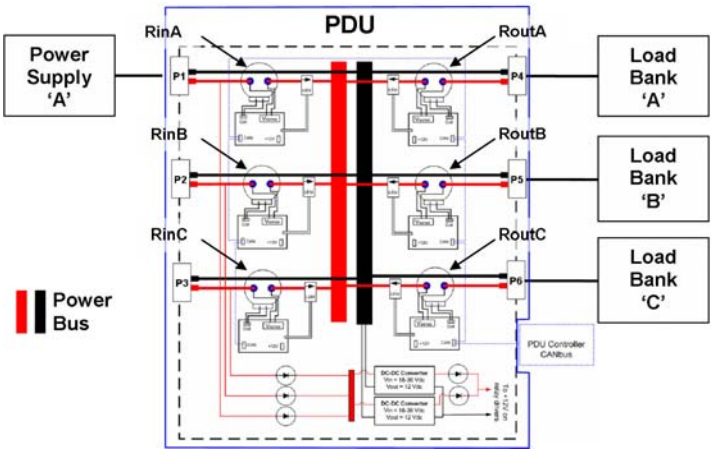


Figure 4.—Schematic for PDU test-bed.

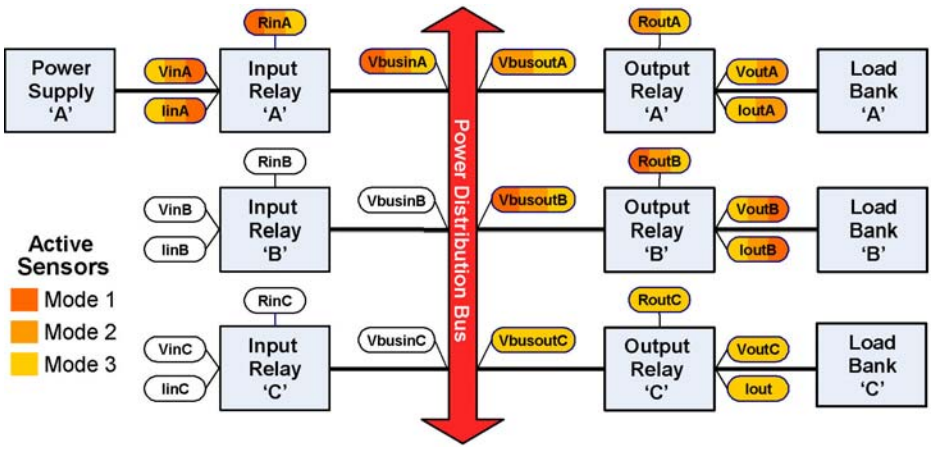


Figure 5.—Diagram showing active relays and sensors for primary PHALT system operating modes.

TABLE 1.—PDU OPERATING MODES FOR TESTING

| | Relays | | | | | |
|---|---|---|---|---|---|---|
| | RinA | RinB | RinC | RoutA | RoutB | RoutC |
| Mode 1 | ON | OFF | OFF | OFF | ON | OFF |
| Mode 2 | ON | OFF | OFF | ON | ON | OFF |

TABLE 2.—DISCRETE VARIABLE DEFINED VALUES FOR SDQ DEMONSTRATION NETWORK

| Configuration | Relay output A current | Relay output B current | Discrete current load |
|---|---|---|---|
| Mode 1 | ----- | 15 A | 0 |
| | ---- | 20 A | 1 |
| Mode 2 | 15 A | 15 A | 0 |
| | 20 A | 15 A | 1 |
| | 15 A | 20 A | 2 |
| | 20 A | 20 A | 3 |

PHALT System connection to the PDU test-bed was accomplished through a CAN bus. Data from each of the relays were sent as separate 64 bit words, each with its own ID. Each word contained values for relay input and output voltages, as well as, values for relay current and an indicator bit (open or closed). The voltages and current values were sent as 12 bit words and converted to double-type real values. The development laptop was connected via the serial cable for both display of select real-time data and for downloading the needed model modifications.

It is also important to note that, for this initial application-specific implementation of the PHALT System, the Simulink diagram shown in figure 3 was replicated once for each sensor in the test-bed. This approach requires the PHALT System software to be reprogrammed if the number of sensors changes. A more general approach is envisioned that does not require reprogramming. However, limitations in some of the built-in Matlab/Simulink functions must be resolved before that approach can be implemented.

## PDU Sensor Data Qualification Network

A sensor qualification network was established for the PDU test-bed described previously. Switch indicators were provided from the test-bed to indicate which relays were closed. Loads were switched in (active load) and out (inactive load) throughout a given test to simulate typical PDU operation. When active, each of the current loads had two operating points, 15 or 20 A. The PDU had two primary operating modes; each mode defined by a unique combination of closed relays. For mode 1, only output relay “B” was closed. For mode 2, output relays A and B were closed. For mode 3 all three output relays were closed. Mode 3 was not available during hardware-in-the-loop tests. The operational modes were

further subdivided by the output current loads. Current output load status was inferred by input and output current measurements. The addition of sub-modes allowed the relationships to be more refined and the associated thresholds to be tighter, thereby reducing the risk of incorrect or missed detections.

The data qualification network was developed by defining constraints (i.e., relationships) between the system parameters. The first group of applied constraints utilized the physical redundancy of the voltage measurements. Due to the configuration of the electrical circuit, the voltage measurements were essentially identical (e.g., VinA = VbusInA). Conservation of charge was used to constrain the input current to the sum of the output currents. Additionally, relationships between the voltages and currents were included into the network of constraints. A description of the network, initially demonstrated off-line using test data playback, is presented in reference 1.

While the hardware test-bed configuration used in the real-time hardware-in-the-loop demonstration was similar to that used to develop the off-line demonstration, there were some differences in operational characteristics due to test-bed modifications. Therefore the network had to be re-trained on nominal data from the reconfigured test-bed in order to provide proper data qualification. This required only a single set of nominal runs to be incorporated into the training data suite.

## Testing and Evaluation Results

Upon completion of the real-time modules of the SDQ algorithms, the PHALT System was used to verify and validate algorithms in a real-time playback mode. This phase 2 testing utilized the same failure data sets used in earlier phase 1 tests. The real-time SDQ algorithms were successfully verified by generating the exact detection times for all 39 fault cases reported in reference 1.

For phase 3 evaluation and testing, the PHALT System was used in a real-time monitoring mode with the PDU test-bed. During this phase, tests were conducted only in operating modes 1 and 2, as operating mode 3 was unavailable. For each test, a simulated fault was superimposed on nominal data obtained from the test-bed. Examples of the simulated faults are shown in figure 6. The DQVS software was used to analyze the resulting data for sensor faults. The analysis process and results from the testing are briefly discussed in this section.

The diagnostic process was as follows. Sensors were sampled at 500 Hz by each relay controller board which also pre-processed the “raw” data and broadcast it to the CAN bus at 25 Hz. The PHALT System read the “raw” data from the CAN bus at the same rate. The “raw” data was converted to engineering units by the PHALT System “Data Conditioning” module. The “Fault Simulation” module then computed a value for a selected fault and superimposed it on the nominal data. The “Diagnostic” module analyzed the “faulty” data using the DQVS algorithms to determine whether or not the sensor had failed. Sensor failures were reported in the form of a log file.

For these tests, the DQVS software required a minimum of 5 relationships to fail at a given point in time before issuing an alarm, and an alarm at three consecutive sample times was required to declare the sensor failed. These requirements for detecting and declaring a sensor failure are determined by the DQVS software based on user-provided sensor design information and system requirements for missed detections and false alarms. Once a sensor was declared failed, all relationships involving that sensor were disregarded by the SDQ algorithms. Following a declared sensor failure, the network performed an internal check to determine if enough valid relationships remained to allow the qualification process to function properly. If there were not enough valid relationships, the SDQ process was terminated. An alternate approach that should be considered for future demonstrations would be to have the system revert to reasonableness checks (i.e., state-of-the-art approach) when the number of valid relationships proved insufficient for qualifying the sensor network.

Table 3 shows results from hardware-in-the-loop testing of the PHALT System with the PDU test-bed. Tests included 13 sensor fault scenarios for mode 1, and 12 fault scenarios for mode 2. These fault scenarios are defined by the “Fault Type” (column three) and the sensor that was failed (column four). Refer to figure 5 for the active sensors and relays associated with each operating mode. A “Pass” in the “Failure Isolated” column indicates that the SDQ system correctly isolated the fault to the failed sensor.

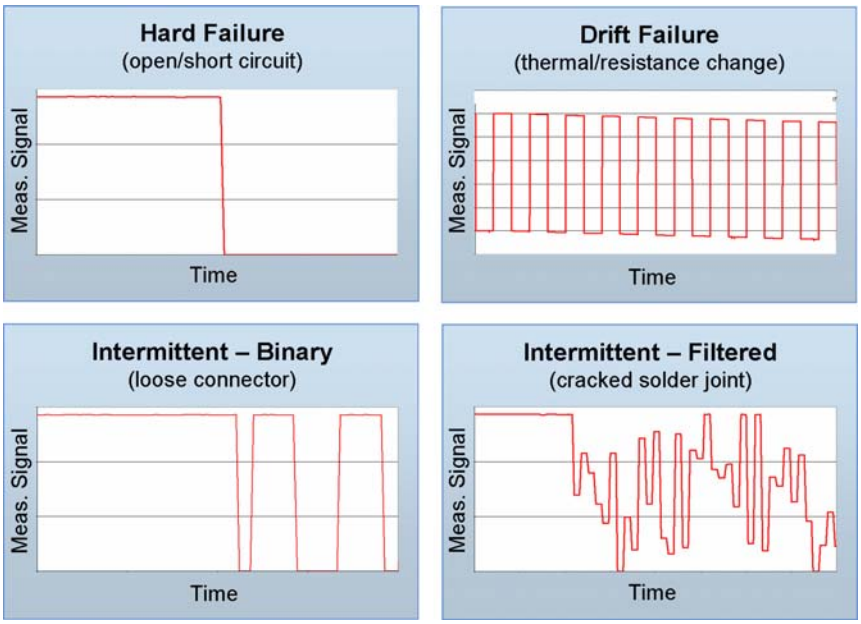


Figure 6.—Examples of simulated sensor faults used in PHALT System demonstration.

TABLE 3.—RESULTS FOR PHALT SYSTEM SDQ TESTS WITH PDU TEST-BED

| Mode | Scenario | Fault type | Sensor failed | Failure isolated | Time to detect (sec) |
|---|---|---|---|---|---|
| Mode 1 | Fault 1 | Intermittent—Binary Mode | VoutB | Pass | 0.08 |
| | Fault 2 | Intermittent—Binary Mode | IoutB | Pass | 0.08 |
| | Fault 3 | Intermittent—Binary Mode | IinA | Pass | 0.08 |
| | Fault 4 | Intermittent—Filtered Mode | VoutB | Pass | 0.08 |
| | Fault 5 | Intermittent—Filtered Mode | IoutB | Pass | 0.08 |
| | Fault 6 | Intermittent—Filtered Mode | IinA | Pass | 0.08 |
| | Fault 7 | Hard Low | VoutB | Pass | 0.08 |
| | Fault 8 | Hard Low | IoutB | Pass | 0.08 |
| | Fault 9 | Hard High | VinA | Pass | 0.08 |
| | Fault 10 | Hard Low | IinA | Pass | 0.08 |
| | Fault 11 | Drift—Low Rate | IoutB | Pass | 74.40 |
| | Fault 12 | Drift—Med Rate | IoutB | Pass | 38.08 |
| | Fault 13 | Drift—High Rate | IoutB | Pass | 26.92 |
| Mode 2 | Fault 1 | Intermittent—Binary Mode | VoutB | Pass | 0.08 |
| | Fault 2 | Intermittent—Binary Mode | IoutB | Pass | 0.08 |
| | Fault 3 | Intermittent—Binary Mode | IinA | Pass | 0.08 |
| | Fault 4 | Intermittent—Filtered Mode | VoutB | Pass | 0.08 |
| | Fault 5 | Intermittent—Filtered Mode | IoutB | Pass | 0.08 |
| | Fault 6 | Intermittent—Filtered Mode | IinA | Pass | 0.08 |
| | Fault 7 | Hard Low | VoutA | Pass | 0.08 |
| | Fault 8 | Hard Low | IoutA | Pass | 0.08 |
| | Fault 9 | Hard High | VinA | Pass | 0.08 |
| | Fault 10 | Hard Low | IinA | Pass | 0.08 |
| | Fault 11 | Drift—Low Rate | IinA | Pass | 194.56 |
| | Fault 12 | Drift—High Rate | IinA | Pass | 52.52 |

As can be seen in the results, the PHALT System demonstrated an ability to correctly detect all the simulated faults. For the hard faults, all declaration times occurred within 0.08 sec of fault occurrence. These results are consistent with those obtained previously using test data playback. The sample rate of the test-bed is 25 Hz; therefore the detection is immediate and consistently within the constraint of three consecutive data cycles. A conventional hard-limit detection system with limits at the sensor range would perform no better given the same requirement for a limit exceedance during three consecutive data cycles. Therefore, these results are comparable to the current state-of-the-art.

Detection times for the intermittent sensor failures were similar to those of the hard faults; they are detected within 0.08 sec. It is important to note here that the intermittent-binary mode faults basically behave the same as the hard faults for the same initial time period. However, the intermittent-filtered mode faults do not. Since they output partial signal values, these faults would not necessarily be detected using a conventional hard-limit detection system.

Drift faults are another category of faults that would not generally be detected by conventional methods, at least not until they reached a hard limit threshold. SDQ methods allow drift faults to be detected much earlier. Detection times for drift failures were dependent upon the fault progression rate and the state of the loads at the time the failure was initiated. Load data used during phases 1 and 2 could not be exactly replicated during phase 3 because of the test-bed reconfiguration, so a direct comparison of the results is not possible. However, all drift failures were correctly identified during phase 3 tests; and trends in the data showed an expected similarity to phase 1 results (ref. 1), that is, time-to-detect was proportional to drift rate.

It is important to note that no real-time constraints were violated during the real-time demonstration. Timing studies showed that the PHALT System software required less than 10 μs to complete its tasks during each 40 ms sample window.

## Summary and Conclusions

In this paper, a PHALT System was presented as a platform for developing and maturing health management algorithms for aerospace systems. The hardware/software platform provides a development environment for initial implementation, and a capability to seamlessly generate code for real-time hardware-in-the-loop tests. A functional architecture was described in some detail, as well as, the subsequent selection and integration of hardware and software needed to implement the architecture. An initial demonstration of the PHALT System was conducted to demonstrate the utility of the approach. For that demonstration, sensor data qualification algorithms were used as the diagnostic tool; and a prototype PDU test-bed was used as the target for a real-time hardware-in-the-loop demonstration. The application-specific implementation and demonstration of the PHALT System were discussed, including results. Test results showed that sensor failures were correctly detected and isolated without violation of any real-time constraints leading to a successful initial demonstration.

## Future Work

NASA is currently working to develop a more complete and more general implementation of the PHALT System software. In its envisioned state, this general implementation would fully realize the functional architecture described in this paper; and would allow the software to be reconfigured via parameter settings rather than requiring significant reprogramming for each new application. Future demonstrations of the PHALT System should also address issues pertinent to systems health management, such as: variability in sensor resolution and sample rate, failures within a closed-loop control system, and system-level diagnostic/prognostic reasoning.

# References


1. Maul, William A.; Melcher, Kevin J.; Chicatelli, Amy; and Sowers, T. Shane, "Sensor Data Qualification for Autonomous Operation of Space Systems," NASA/TM—2006–214475, November 2006.
2. Bickford, R.L.; Bickmore, T.W.; and Caluori, V.A: "Real-time Sensor Validation for Autonomous Flight Control," AIAA/ASME/SAE/ASEE 33rd Joint Propulsion Conference and Exhibit, AIAA–1997–2901, July 1997.
3. Bickford, R.L.; Bickmore, T.W.; Meyer, C.M.; and Zakrajsek, J.F.: "Real-time Sensor Validation for Propulsion Systems," AIAA Defense and Civil Space Programs Conference and Exhibit, Collection of Technical Papers (A98-45901 12-66), AIAA–1998–5184, October 1998.
4. Bickford, R.L.; Bickmore, T.W.; Meyer, C.M.; and Zakrajsek, J.F.: Real-time Sensor Data Validation for Space Shuttle Main Engine Telemetry Monitoring," AIAA/ASME/SAE/ASEE 35th Joint Propulsion Conference and Exhibit, AIAA–1999–2531, June 1999.
5. Herzog, J.P.; Yue, Y.Y.; and Bickford, R.L.: "Dynamics Sensor Validation for Reusable Launch Vehicle Propulsion," AIAA/ASME/SAE/ASEE 34th Joint Propulsion Conference and Exhibit, AIAA–1998–3604, July 1998.
6. Hines, J.; and Davis, E.: "Lessons Learned from the U.S. Nuclear Power Plant On-Line Monitoring Programs," Progress in Nuclear Energy, vol. 46, no. 3–4, pp. 176–189, Elsevier, 2004.

| REPORT DOCUMENTATION PAGE | Form Approved OMB No. 0704-0188 |
|---|---|



| 1. REPORT DATE *(DD-MM-YYYY)* | 2. REPORT TYPE | 3. DATES COVERED *(From - To)* |
|---|---|---|
| 01-07-2007 | Technical Memorandum | |

**4. TITLE AND SUBTITLE**
Development and Application of a Portable Health Algorithms Test System

**5a. CONTRACT NUMBER**

**5b. GRANT NUMBER**

**5c. PROGRAM ELEMENT NUMBER**

**6. AUTHOR(S)**
Melcher, Kevin, J.; Fulton, Christopher, E.; Maul, William, A.; Sowers, T., Shane

**5d. PROJECT NUMBER**

**5e. TASK NUMBER**

**5f. WORK UNIT NUMBER**
WBS 136905.08.05.08.08.01.03

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
National Aeronautics and Space Administration
John H. Glenn Research Center at Lewis Field
Cleveland, Ohio 44135-3191

**8. PERFORMING ORGANIZATION REPORT NUMBER**
E-16055

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
National Aeronautics and Space Administration
Washington, DC 20546-0001

**10. SPONSORING/MONITORS ACRONYM(S)**
NASA

**11. SPONSORING/MONITORING REPORT NUMBER**
NASA/TM-2007-214840

**12. DISTRIBUTION/AVAILABILITY STATEMENT**
Unclassified-Unlimited
Subject Category: 18
Available electronically at http://gltrs.grc.nasa.gov
This publication is available from the NASA Center for AeroSpace Information, 301-621-0390

**13. SUPPLEMENTARY NOTES**

**14. ABSTRACT**
This paper describes the development and initial demonstration of a Portable Health Algorithms Test (PHALT) System that is being developed by researchers at the NASA Glenn Research Center (GRC). The PHALT System was conceived as a means of evolving the maturity and credibility of algorithms developed to assess the health of aerospace systems. Comprising an integrated hardware-software environment, the PHALT System allows systems health management algorithms to be developed in a graphical programming environment; to be tested and refined using system simulation or test data playback; and finally, to be evaluated in a real-time hardware-in-the-loop mode with a live test article. In this paper, PHALT System development is described through the presentation of a functional architecture, followed by the selection and integration of hardware and software. Also described is an initial real-time hardware-in-the-loop demonstration that used sensor data qualification algorithms to diagnose and isolate simulated sensor failures in a prototype Power Distribution Unit test-bed. Success of the initial demonstration is highlighted by the correct detection of all sensor failures and the absence of any real-time constraint violations.

**15. SUBJECT TERMS**
Systems health monitoring; Fault detection

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | c. THIS PAGE | | | Kevin J. Melcher |
| U | U | U | | 18 | 19b. TELEPHONE NUMBER *(include area code)* 216-433-3743 |